|  |
| --- |
|  |
| AUTOMATIC LICENSE PLATE RECOGNITION PROJECT CODE |
|  |
| **By****Anthony Tierney** |
|  |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |
| AUTOMATIC LICENSE PLATE RECOGNITION PROJECT CODEAnthony TierneyI.T. Carlow, Kilkenny Road, Carlow |
| 28-Apr-14 |
|  |

|  |
| --- |
|  |

Table of Contents

1. MainActivity.java 1

2. ALPRCamera.java 8

3. CameraPreview.java 14

4. Image.java 23

5. CRUDCarPark.java 34

6. CRUDUsers.java 46

7. NewVehicle.java 60

8. ConvolutionMatrix.java 67

# MainActivity.java

package com.alpr.alpreire;

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStream;

import java.io.InputStreamReader;

import java.net.HttpURLConnection;

import java.net.URL;

import android.app.Activity;

import android.content.Context;

import android.content.Intent;

import android.os.Bundle;

import android.os.StrictMode;

import android.text.Editable;

import android.view.View;

import android.view.Window;

import android.view.WindowManager;

import android.widget.Button;

import android.widget.EditText;

import android.widget.Toast;

public class MainActivity extends Activity {

 Context ctx;

 @Override

 public void onCreate(Bundle savedInstanceState) {

 super.onCreate(savedInstanceState);

 ctx = this;

 requestWindowFeature(Window.FEATURE\_NO\_TITLE);

 getWindow().addFlags(WindowManager.LayoutParams.FLAG\_FULLSCREEN);

 setContentView(R.layout.main);

 performIntent(this);

 StrictMode.ThreadPolicy policy = new StrictMode.

 ThreadPolicy.Builder().permitAll().build();

 StrictMode.setThreadPolicy(policy);

 }

 private void performIntent(Context context){

 final Intent intent = new Intent(context, ALPRCamera.class);

 Button loginBtn = (Button) findViewById(R.id.loginButton);

 loginBtn.setOnClickListener(new View.OnClickListener() {

 @Override

 public void onClick(View arg0) {

 EditText et = (EditText)findViewById(R.id.loginuser);

 Editable userid = et.getText();

 et = (EditText)findViewById(R.id.loginpasswd);

 Editable passwd = et.getText();

 try {

 String result = doLogin(userid.toString(), passwd.toString());

 if ( result.equals("True")){

 startActivity(intent);

 }

 else{

 Toast toast = Toast.makeText(ctx, "Login failed", Toast.LENGTH\_SHORT);

 toast.show();

 }

 } catch (IOException e) {

 // TODO Auto-generated catch block

 e.printStackTrace();

 }

 }

 });

 }

 private String doLogin(String username, String passwd) throws IOException{

 InputStream is = null;

 // Only display the first 500 characters of the retrieved

 // web page content.

 try {

 URL url = new URL("http://alpreire.appspot.com/login/admin/admin");

 HttpURLConnection conn = (HttpURLConnection) url.openConnection();

 conn.setReadTimeout(10000 /\* milliseconds \*/);

 conn.setConnectTimeout(15000 /\* milliseconds \*/);

 conn.setRequestMethod("GET");

 conn.setDoInput(true);

 // Starts the query

 conn.connect();

 //int response = conn.getResponseCode();

 //Log.d(DEBUG\_TAG, "The response is: " + response);

 is = conn.getInputStream();

 // Convert the InputStream into a string

 String contentAsString = readStream(is);

 return contentAsString;

 // Makes sure that the InputStream is closed after the app is

 // finished using it.

 } finally {

 if (is != null) {

 is.close();

 }

 }

 private String readStream(InputStream in) {

 BufferedReader reader = null;

 StringBuilder sb = new StringBuilder();

 try {

 reader = new BufferedReader(new InputStreamReader(in));

 String line = "";

 while ((line = reader.readLine()) != null) {

 sb.append(line);

 }

 } catch (IOException e) {

 e.printStackTrace();

 } finally {

 if (reader != null) {

 try {

 reader.close();

 } catch (IOException e) {

 e.printStackTrace();

 }

 }

 }

 return sb.toString();

 }

}

# ALPRCamera.java

package com.alpr.alpreire;

import android.app.Activity;

import android.content.Context;

import android.content.Intent;

import android.hardware.Camera;

import android.hardware.Camera.PictureCallback;

import android.hardware.Camera.ShutterCallback;

import android.os.Bundle;

import android.view.Menu;

import android.view.MenuItem;

import android.view.SurfaceView;

import android.view.View;

import android.view.View.OnClickListener;

import android.widget.Button;

import android.widget.FrameLayout;

import android.widget.Toast;

public class ALPRCamera extends Activity {

 //private static final String TAG = "CamTestActivity";

 CameraPreview preview;

 Button cameraShutter;

 Camera camera;

 String fileName;

 Activity act;

 Context ctx;

 byte[] image;

 @Override

 protected void onCreate(Bundle savedInstanceState) {

 super.onCreate(savedInstanceState);

 setContentView(R.layout.activity\_alprcamera);

 // Show the Up button in the action bar.

 //setupActionBar();

 ctx = this;

 preview = new CameraPreview(this, (SurfaceView) findViewById(R.id.surfaceView));

 // preview.setLayoutParams(new LayoutParams(LayoutParams.FILL\_PARENT,

 // LayoutParams.FILL\_PARENT));

 ((FrameLayout) findViewById(R.id.preview)).addView(preview);

 preview.setKeepScreenOn(true);

 onClicks();

 }

 private void onClicks(){//handle all the onClick methods

 cameraShutter = (Button) findViewById(R.id.buttonClick);

 cameraShutter.setOnClickListener(new OnClickListener() {

 @Override

 public void onClick(View v) {

 preview.mCamera.takePicture(shutterCallback, rawCallback,

 jpegCallback);

 camera.takePicture(shutterCallback, rawCallback, jpegCallback);

 }

 });

// cameraShutter.setOnLongClickListener(new OnLongClickListener() {

// @Override

// public boolean onLongClick(View arg0) {

// camera.autoFocus(new AutoFocusCallback() {

// @Override

// public void onAutoFocus(boolean arg0, Camera arg1) {

// // camera.takePicture(shutterCallback, rawCallback, jpegCallback);

// }

// });

// return true;

// }

// });

 }

 @Override

 public boolean onCreateOptionsMenu(Menu menu) {

 // Inflate the menu; this adds items to the action bar if it is present.

 getMenuInflater().inflate(R.menu.alprcamera, menu);

 return true;

 }

 @Override

 public boolean onOptionsItemSelected(MenuItem item) {

 switch (item.getItemId()) {

 case android.R.id.home:

 //NavUtils.navigateUpFromSameTask(this);

 return true;

 case R.id.CRUDCarPark:

 Intent cpintent = new Intent(this, CRUDCarpark.class);

 startActivity(cpintent);

 return true;

 case R.id.CRUDUsers:

 Intent i = new Intent(this, CRUDUsers.class);

 startActivity(i);

 default:

 return super.onOptionsItemSelected(item);

 }

 //return super.onOptionsItemSelected(item);

 }

 @Override

 protected void onResume() {

 super.onResume();

 // preview.camera = Camera.open();

 camera = Camera.open();

 camera.startPreview();

 preview.setCamera(camera);

 }

 @Override

 protected void onPause() {

 if (camera != null) {

 camera.stopPreview();

 preview.setCamera(null);

 camera.release();

 camera = null;

 }

 super.onPause();

 }

 private void resetCam() {

 camera.startPreview();

 preview.setCamera(camera);

 }

 ShutterCallback shutterCallback = new ShutterCallback() {

 @Override

 public void onShutter() {

 // Log.d(TAG, "onShutter'd");

 }

 };

 PictureCallback rawCallback = new PictureCallback() {

 @Override

 public void onPictureTaken(byte[] data, Camera camera) {

 // Log.d(TAG, "onPictureTaken - raw");

 }

 };

 PictureCallback jpegCallback = new PictureCallback() {

 @Override

 public void onPictureTaken(byte[] data, Camera camera) {

 try {

 String imageKey = "imageByte";

 Intent picTakenIntent = new Intent(ctx, Image.class);

 //Bundle extras = new Bundle();

 //extras.putByteArray(imageKey, data);

 picTakenIntent.putExtra(imageKey, data);

 startActivity(picTakenIntent);

 //resetCam();

 } catch (Exception e) {

 Toast toast = Toast.makeText(ctx, "Failed to process image", Toast.LENGTH\_SHORT);

 toast.show();

 resetCam();

 }

 }

 };

}

# CameraPreview.java

package com.alpr.alpreire;

import java.io.IOException;

import java.util.List;

import android.content.Context;

import android.hardware.Camera;

import android.hardware.Camera.Size;

import android.util.Log;

import android.view.SurfaceHolder;

import android.view.SurfaceView;

import android.view.View;

import android.view.ViewGroup;

class CameraPreview extends ViewGroup implements SurfaceHolder.Callback {

 private final String TAG = "Preview";

 SurfaceView mSurfaceView;

 SurfaceHolder mHolder;

 Size mPreviewSize;

 List<Size> mSupportedPreviewSizes;

 Camera mCamera;

 List<Camera.Size> mSupportedMP;

 CameraPreview(Context context, SurfaceView sv) {

 super(context);

 mSurfaceView = sv;

// addView(mSurfaceView);

 mHolder = mSurfaceView.getHolder();

 mHolder.addCallback(this);

 //mHolder.setType(SurfaceHolder.SURFACE\_TYPE\_PUSH\_BUFFERS);

 }

 public void setCamera(Camera camera) {

 mCamera = camera;

 if (mCamera != null) {

 mSupportedPreviewSizes = mCamera.getParameters().getSupportedPreviewSizes();

 requestLayout();

 //get supported camera resolutions

 mSupportedMP = mCamera.getParameters().getSupportedPictureSizes();

 Size mp;

 mp = mSupportedMP.get(mSupportedMP.size() - 2);

 // get Camera parameters

 Camera.Parameters params = mCamera.getParameters();

 mCamera.setDisplayOrientation(90);

 List<String> focusModes = params.getSupportedFocusModes();

 if (focusModes.contains(Camera.Parameters.FOCUS\_MODE\_CONTINUOUS\_PICTURE)) {

 //set mp

 params.setPictureSize(mp.width, mp.height);

 // set the focus mode

 params.setFocusMode(Camera.Parameters.FOCUS\_MODE\_CONTINUOUS\_PICTURE);

 //params.setFocusMode(Camera.Parameters.FOCUS\_MODE\_AUTO);;

 //params.setFlashMode(Camera.Parameters.FLASH\_MODE\_AUTO);

 // set Camera parameters

 mCamera.setParameters(params);

 }

 }

 }

 @Override

 protected void onMeasure(int widthMeasureSpec, int heightMeasureSpec) {

 // We purposely disregard child measurements because act as a

 // wrapper to a SurfaceView that centers the camera preview instead

 // of stretching it.

 final int width = resolveSize(getSuggestedMinimumWidth(), widthMeasureSpec);

 final int height = resolveSize(getSuggestedMinimumHeight(), heightMeasureSpec);

 setMeasuredDimension(width, height);

 if (mSupportedPreviewSizes != null) {

 mPreviewSize = getOptimalPreviewSize(mSupportedPreviewSizes, width, height);

 }

 }

 @Override

 protected void onLayout(boolean changed, int l, int t, int r, int b) {

 if (changed && getChildCount() > 0) {

 final View child = getChildAt(0);

 final int width = r - l;

 final int height = b - t;

 int previewWidth = width;

 int previewHeight = height;

 if (mPreviewSize != null) {

 previewWidth = mPreviewSize.width;

 previewHeight = mPreviewSize.height;

 }

 // Center the child SurfaceView within the parent.

 if (width \* previewHeight > height \* previewWidth) {

 final int scaledChildWidth = previewWidth \* height / previewHeight;

 child.layout((width - scaledChildWidth) / 2, 0,

 (width + scaledChildWidth) / 2, height);

 } else {

 final int scaledChildHeight = previewHeight \* width / previewWidth;

 child.layout(0, (height - scaledChildHeight) / 2,

 width, (height + scaledChildHeight) / 2);

 }

 }

 }

 @Override

 public void surfaceCreated(SurfaceHolder holder) {

 // The Surface has been created, acquire the camera and tell it where

 // to draw.

 try {

 if (mCamera != null) {

 mCamera.setPreviewDisplay(holder);

 }

 } catch (IOException exception) {

 Log.e(TAG, "IOException caused by setPreviewDisplay()", exception);

 }

 }

 @Override

 public void surfaceDestroyed(SurfaceHolder holder) {

 // Surface will be destroyed when we return, so stop the preview.

 if (mCamera != null) {

 mCamera.stopPreview();

 }

 }

 private Size getOptimalPreviewSize(List<Size> sizes, int w, int h) {

 final double ASPECT\_TOLERANCE = 0.1;

 double targetRatio = (double) w / h;

 if (sizes == null) return null;

 Size optimalSize = null;

 double minDiff = Double.MAX\_VALUE;

 int targetHeight = h;

 // Try to find an size match aspect ratio and size

 for (Size size : sizes) {

 double ratio = (double) size.width / size.height;

 if (Math.abs(ratio - targetRatio) > ASPECT\_TOLERANCE) continue;

 if (Math.abs(size.height - targetHeight) < minDiff) {

 optimalSize = size;

 minDiff = Math.abs(size.height - targetHeight);

 }

 }

 // Cannot find the one match the aspect ratio, ignore the requirement

 if (optimalSize == null) {

 minDiff = Double.MAX\_VALUE;

 for (Size size : sizes) {

 if (Math.abs(size.height - targetHeight) < minDiff) {

 optimalSize = size;

 minDiff = Math.abs(size.height - targetHeight);

 }

 }

 }

 return optimalSize;

 }

 @Override

 public void surfaceChanged(SurfaceHolder holder, int format, int w, int h) {

 if(mCamera != null) {

 Camera.Parameters parameters = mCamera.getParameters();

 parameters.setPreviewSize(mPreviewSize.width, mPreviewSize.height);

 requestLayout();

 mCamera.setParameters(parameters);

 mCamera.startPreview();

 }

 }

}

# Image.java

package com.alpr.alpreire;

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStream;

import java.io.InputStreamReader;

import java.net.HttpURLConnection;

import java.net.URL;

import android.app.Activity;

import android.app.AlertDialog;

import android.content.DialogInterface;

import android.graphics.Bitmap;

import android.graphics.BitmapFactory;

import android.graphics.Canvas;

import android.graphics.Color;

import android.graphics.ColorMatrix;

import android.graphics.ColorMatrixColorFilter;

import android.graphics.Paint;

import android.media.ThumbnailUtils;

import android.os.Bundle;

import android.text.Editable;

import android.view.Menu;

import android.view.MenuItem;

import android.widget.CheckBox;

import android.widget.CompoundButton;

import android.widget.CompoundButton.OnCheckedChangeListener;

import android.widget.EditText;

import android.widget.ImageView;

import android.widget.LinearLayout;

import android.widget.Toast;

import com.alpr.alpreire.imagemanipulation.ConvolutionMatrix;

public class Image extends Activity {

 private Bitmap dstBmp;

 @Override

 protected void onCreate(Bundle savedInstanceState) {

 super.onCreate(savedInstanceState);

 setContentView(R.layout.activity\_image);

 // Show the Up button in the action bar.

 //setupActionBar();

// if (!OpenCVLoader.initAsync(OpenCVLoader.OPENCV\_VERSION\_2\_4\_2, this, mOpenCVCallBack))

// {

// Toast tst = null;

// tst = Toast.makeText(this, "Cannot connect to OpenCV Manager", Toast.LENGTH\_SHORT);

// tst.show();

//

// }

// else {

 byte[] imageData = getIntent().getExtras().getByteArray("imageByte");

 Bitmap srcBmp = BitmapFactory.decodeByteArray(imageData, 0, imageData.length);

 this.dstBmp = null;

 this.dstBmp = ThumbnailUtils.extractThumbnail(srcBmp, 90, 600);

 processImage();

// ImageView imageView = (ImageView) findViewById(R.id.setImage);

// imageView.setImageBitmap(this.dstBmp);

 //}

 }

 private void processImage(){

 ImageView imageView = (ImageView) findViewById(R.id.setImageGrey);

 toGrayScale();

 imageView.setImageBitmap(this.dstBmp);

 imageView = (ImageView) findViewById(R.id.setImageBlur );

 applyGaussianBlur();

 imageView.setImageBitmap(this.dstBmp);

 imageView = (ImageView) findViewById(R.id.setImageThreshold);

 imageThreshold();

 imageView.setImageBitmap(this.dstBmp);

 imageView = (ImageView) findViewById(R.id.setImageEdge);

 edgeDetect();

 imageView.setImageBitmap(this.dstBmp);

 templateMatch();

 }

 @Override

 public boolean onCreateOptionsMenu(Menu menu) {

 // Inflate the menu; this adds items to the action bar if it is present.

 getMenuInflater().inflate(R.menu.image, menu);

 return true;

 }

 private Editable regPlate;

 @Override

 public boolean onOptionsItemSelected(MenuItem item) {

 switch (item.getItemId()) {

 case android.R.id.home:

 // This ID represents the Home or Up button. In the case of this

 // activity, the Up button is shown. Use NavUtils to allow users

 // to navigate up one level in the application structure. For

 // more details, see the Navigation pattern on Android Design:

 //

 // http://developer.android.com/design/patterns/navigation.html#up-vs-back

 //

 //NavUtils.navigateUpFromSameTask(this);

 return true;

 case R.id.manEntry:

 AlertDialog.Builder builder= new AlertDialog.Builder(this);

 builder.setTitle("Manual Entry").setMessage("Enter the Licence plate number:");

 //TextView licencePlateNum = new TextView(this);

 final EditText editText = new EditText(this);

 final CheckBox cb = new CheckBox(this);

 cb.setText("Parked Legally?");

 cb.setOnCheckedChangeListener(new OnCheckedChangeListener(){

 @Override

 public void onCheckedChanged(CompoundButton chkBtn, boolean isChecked) {

 // TODO Auto-generated method stub

 //chkBtn.setChecked(!isChecked);

 }

 });

 LinearLayout linLay = new LinearLayout(this);

 editText.setWidth(450);

 linLay.addView(editText);

 cb.setChecked(true);

 linLay.addView(cb);

 builder.setView(linLay);

 builder.setPositiveButton("OK", new DialogInterface.OnClickListener() {

 public void onClick(DialogInterface dialog, int id) {

 // User clicked OK button

 regPlate= editText.getText();

 try {

 String carCheck=getCloudContent("http://alpreire.appspot.com/checkcar/"+regPlate.toString());

 if(carCheck != null){

 Toast toast = Toast.makeText(getApplicationContext(), carCheck, Toast.LENGTH\_SHORT);

 toast.show();

 }

 } catch (IOException e) {

 // TODO Auto-generated catch block

 e.printStackTrace();

 }

 }

 });

 builder.setNegativeButton("Cancel", new DialogInterface.OnClickListener() {

 public void onClick(DialogInterface dialog, int id) {

 // User cancelled the dialog

 }

 });

 AlertDialog dialog = builder.create();

 dialog.show();

 }

 return super.onOptionsItemSelected(item);

 }

// private BaseLoaderCallback mOpenCVCallBack = new BaseLoaderCallback(this) {

// @Override

// public void onManagerConnected(int status) {

// switch (status) {

// case LoaderCallbackInterface.SUCCESS:

// {

// //processImage(dstBmp);

// } break;

// default:

// {

// super.onManagerConnected(status);

// } break;

// }

// }

// };

 private void toGrayScale()

 {

 Bitmap bmpGrayscale = Bitmap.createBitmap(this.dstBmp.getWidth(), this.dstBmp.getHeight(), Bitmap.Config.RGB\_565);

 Canvas c = new Canvas(bmpGrayscale);

 Paint paint = new Paint();

 ColorMatrix cm = new ColorMatrix();

 cm.setSaturation(0);

 ColorMatrixColorFilter f = new ColorMatrixColorFilter(cm);

 paint.setColorFilter(f);

 c.drawBitmap(this.dstBmp, 0, 0, paint);

 this.dstBmp=bmpGrayscale;

 }

 private void applyGaussianBlur() {

 //set gaussian blur configuration

 double[][] GaussianBlurConfig = new double[][] {

 { 1, 2, 1 },

 { 2, 4, 2 },

 { 1, 2, 1 }

 };

 // create instance of Convolution matrix

 ConvolutionMatrix convMatrix = new ConvolutionMatrix(3);

 // Apply Configuration

 convMatrix.applyConfig(GaussianBlurConfig);

 convMatrix.Factor = 16;

 convMatrix.Offset = 0;

 //return out put bitmap

 this.dstBmp = ConvolutionMatrix.computeConvolution3x3(this.dstBmp, convMatrix);

 }

 private void imageThreshold(){

 final int threshold = 127;

 int red = 0, blue = 0, green = 0;

 int height = this.dstBmp.getHeight();

 int width = this.dstBmp.getWidth();

 int pixel = 0;

 for(int x = 0;x < width; x++){

 for(int y = 0; y < height; y++){

 pixel = this.dstBmp.getPixel(x, y);

 red = Color.red(pixel);

 blue = Color.blue(pixel);

 green = Color.green(pixel);

 if((red+green+blue)/3 < threshold){

 this.dstBmp.setPixel(x, y, Color.BLACK);

 }

 else{

 this.dstBmp.setPixel(x, y, Color.WHITE);

 }

 }

 }

 }

 private void edgeDetect(){

 double[][] edgeDetectConfig = new double[][] {

 { 0, 1, 0 },

 { 1, -4, 1 },

 { 0, 1, 0 }

 };

 // create instance of Convolution matrix

 ConvolutionMatrix convMatrix = new ConvolutionMatrix(3);

 // Apply Configuration

 convMatrix.applyConfig(edgeDetectConfig);

 convMatrix.Factor = 1;

 convMatrix.Offset = 0;

 //return out put bitmap

 this.dstBmp = ConvolutionMatrix.computeConvolution3x3(this.dstBmp, convMatrix);

 }

 private void templateMatch(){

// System.out.println("\nRunning Template Matching");

// Bitmap num1 = BitmapFactory.decodeResource(getResources(), R.drawable.num1);

// Mat img = new Mat(dstBmp.getWidth(), dstBmp.getHeight(), CvType.CV\_32FC1);

// Utils.bitmapToMat(dstBmp, img);

// Mat templ =new Mat(dstBmp.getWidth(), dstBmp.getHeight(), CvType.CV\_32FC1);

// Utils.bitmapToMat(num1,templ);

// int match\_method = Imgproc.TM\_CCOEFF;

// // / Create the result matrix

// int result\_cols = img.cols() - templ.cols() + 1;

// int result\_rows = img.rows() - templ.rows() + 1;

// Mat result = new Mat(result\_rows, result\_cols, CvType.CV\_32FC1);

//

// // / Do the Matching and Normalize

// Imgproc.matchTemplate(img, templ, result, match\_method);

// Core.normalize(result, result, 0, 1, Core.NORM\_MINMAX, -1, new Mat());

//

// // / Localizing the best match with minMaxLoc

// MinMaxLocResult mmr = Core.minMaxLoc(result);

//

// Point matchLoc;

// if (match\_method == Imgproc.TM\_SQDIFF || match\_method == Imgproc.TM\_SQDIFF\_NORMED) {

// matchLoc = mmr.minLoc;

// } else {

// matchLoc = mmr.maxLoc;

// }

 // Save the visualized detection.

 // System.out.println("Writing "+ outFile);

 // Highgui.imwrite(outFile, img);

 }

 private String getCloudContent(String add) throws IOException {

 InputStream is = null;

 try {

 URL url = new URL(add);

 HttpURLConnection conn = (HttpURLConnection) url.openConnection();

 conn.setReadTimeout(10000 /\* milliseconds \*/);

 conn.setConnectTimeout(15000 /\* milliseconds \*/);

 conn.setRequestMethod("GET");

 conn.setDoInput(true);

 // Starts the query

 conn.connect();

 is = conn.getInputStream();

 // Convert the InputStream into a string

 String contentAsString = readStream(is);

 return contentAsString;

 // Makes sure that the InputStream is closed after the app is

 // finished using it.

 }catch(IOException e){

 Toast tst = Toast.makeText(this, "Ensure that the device is connected to the internet", Toast.LENGTH\_SHORT);

 tst.show();

 }

 finally {

 if (is != null) {

 is.close();

 }

 }

 return null;

 }

 private String readStream(InputStream in) {

 BufferedReader reader = null;

 StringBuilder sb = new StringBuilder();

 try {

 reader = new BufferedReader(new InputStreamReader(in));

 String line = "";

 while ((line = reader.readLine()) != null) {

 sb.append(line);

 }

 } catch (IOException e) {

 e.printStackTrace();

 } finally {

 if (reader != null) {

 try {

 reader.close();

 } catch (IOException e) {

 e.printStackTrace();

 }

 }

 }

 return sb.toString();

 }

}

# CRUDCarPark.java

package com.alpr.alpreire;

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStream;

import java.io.InputStreamReader;

import java.net.HttpURLConnection;

import java.net.URL;

import android.app.Activity;

import android.content.Context;

import android.content.Intent;

import android.os.Bundle;

import android.os.StrictMode;

import android.support.v4.app.NavUtils;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.view.View.OnClickListener;

import android.widget.AdapterView;

import android.widget.AdapterView.OnItemSelectedListener;

import android.widget.ArrayAdapter;

import android.widget.Button;

import android.widget.EditText;

import android.widget.Spinner;

import android.widget.Toast;

public class CRUDCarpark extends Activity implements OnItemSelectedListener {

 private Spinner cpSpinner;

 private int currSpinnerPos;

 private Context ctx = this;

 @Override

 protected void onCreate(Bundle savedInstanceState) {

 super.onCreate(savedInstanceState);

 setContentView(R.layout.activity\_crudcarpark);

 // Show the Up button in the action bar.

 setupActionBar();

 StrictMode.ThreadPolicy policy = new StrictMode.

 ThreadPolicy.Builder().permitAll().build();

 StrictMode.setThreadPolicy(policy);

 String[] names={};

 try {

 names = getCloudContent("http://alpreire.appspot.com/getCarParkNames").split("/");

 } catch (IOException e) {

 e.printStackTrace();

 }

 cpSpinner = (Spinner) findViewById(R.id.carparkspinner);

 ArrayAdapter<String> spinnerAdapter = new ArrayAdapter<String>(this,

 R.layout.simple\_spinner\_item, names);

 // spinnerAdapter.setDropDownViewResource(R.id.userspinner);

 cpSpinner.setAdapter(spinnerAdapter);

 cpSpinner.setSelection(0);

 cpSpinner.setOnItemSelectedListener(this);

 setupClicks();

 }

 private void setupClicks(){

 Button saveBtn = (Button)findViewById(R.id.saveCpBtn);

 saveBtn.setOnClickListener(new OnClickListener(){

 @Override

 public void onClick(View arg0) {

 // TODO Auto-generated method stub

 }

 });

 Button deleteBtn = (Button)findViewById(R.id.deleteCPBtn);

 deleteBtn.setOnClickListener(new OnClickListener(){

 @Override

 public void onClick(View arg0) {

 // TODO Auto-generated method stub

 }

 });

 Button newVehBtn = (Button)findViewById(R.id.newVehBtn);

 newVehBtn.setOnClickListener(new OnClickListener(){

 @Override

 public void onClick(View arg0) {

 // TODO Auto-generated method stub

 Intent intent = new Intent(ctx, NewVehicle.class);

 intent.putExtra("cpName", (String)cpSpinner.getItemAtPosition(currSpinnerPos));

 startActivity(intent);

 }

 });

 }

 /\*\*

 \* Set up the {@link android.app.ActionBar}.

 \*/

 private void setupActionBar() {

 getActionBar().setDisplayHomeAsUpEnabled(true);

 }

 @Override

 public boolean onCreateOptionsMenu(Menu menu) {

 // Inflate the menu; this adds items to the action bar if it is present.

 getMenuInflater().inflate(R.menu.crudcarpark, menu);

 return true;

 }

 @Override

 public boolean onOptionsItemSelected(MenuItem item) {

 switch (item.getItemId()) {

 case android.R.id.home:

 // This ID represents the Home or Up button. In the case of this

 // activity, the Up button is shown. Use NavUtils to allow users

 // to navigate up one level in the application structure. For

 // more details, see the Navigation pattern on Android Design:

 //

 // http://developer.android.com/design/patterns/navigation.html#up-vs-back

 //

 NavUtils.navigateUpFromSameTask(this);

 return true;

 }

 return super.onOptionsItemSelected(item);

 }

 private String getCloudContent(String add) throws IOException {

 InputStream is = null;

 try {

 URL url = new URL(add);

 HttpURLConnection conn = (HttpURLConnection) url.openConnection();

 conn.setReadTimeout(10000 /\* milliseconds \*/);

 conn.setConnectTimeout(15000 /\* milliseconds \*/);

 conn.setRequestMethod("GET");

 conn.setDoInput(true);

 // Starts the query

 conn.connect();

 is = conn.getInputStream();

 // Convert the InputStream into a string

 String contentAsString = readStream(is);

 return contentAsString;

 // Makes sure that the InputStream is closed after the app is

 // finished using it.

 }catch(IOException e){

 Toast tst = Toast.makeText(this, "Ensure that the device is connected to the internet", Toast.LENGTH\_SHORT);

 tst.show();

 }

 finally {

 if (is != null) {

 is.close();

 }

 }

 return null;

 }

 private String readStream(InputStream in) {

 BufferedReader reader = null;

 StringBuilder sb = new StringBuilder();

 try {

 reader = new BufferedReader(new InputStreamReader(in));

 String line = "";

 while ((line = reader.readLine()) != null) {

 sb.append(line);

 }

 } catch (IOException e) {

 e.printStackTrace();

 } finally {

 if (reader != null) {

 try {

 reader.close();

 } catch (IOException e) {

 e.printStackTrace();

 }

 }

 }

 return sb.toString();

 }

 @Override

 public void onItemSelected(AdapterView<?> parent, View viwe, int pos,

 long id) {

 String cpname = (String) parent.getItemAtPosition(pos);

 currSpinnerPos=pos;

 String[] content={};

 try {

 content = getCloudContent("http://alpreire.appspot.com/getCPContent/" + cpname).split("/");

 EditText et = (EditText) findViewById(R.id.cpName);

 et.setText(content[0]);

 et = (EditText) findViewById(R.id.cpAddress);

 et.setText(content[1]);

 et = (EditText) findViewById(R.id.cpCapacity);

 et.setText(content[2]);

 et = (EditText) findViewById(R.id.cpAttendant);

 et.setText(content[3]);

 } catch (IOException e) {

 // TODO Auto-generated catch block

 e.printStackTrace();

 }

 }

 @Override

 public void onNothingSelected(AdapterView<?> arg0) {

 // TODO Auto-generated method stub

 }

}

# CRUDUsers.java

package com.alpr.alpreire;

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStream;

import java.io.InputStreamReader;

import java.net.HttpURLConnection;

import java.net.URL;

import android.app.Activity;

import android.content.Context;

import android.os.Bundle;

import android.os.StrictMode;

import android.support.v4.app.NavUtils;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.widget.AdapterView;

import android.widget.AdapterView.OnItemSelectedListener;

import android.widget.ArrayAdapter;

import android.widget.Button;

import android.widget.EditText;

import android.widget.Spinner;

import android.widget.Toast;

public class CRUDUsers extends Activity implements OnItemSelectedListener {

 private Spinner userSpinner;

 private Context ctx = this;

 @Override

 protected void onCreate(Bundle savedInstanceState) {

 super.onCreate(savedInstanceState);

 setContentView(R.layout.activity\_user\_mod\_admin);

 StrictMode.ThreadPolicy policy = new StrictMode.

 ThreadPolicy.Builder().permitAll().build();

 StrictMode.setThreadPolicy(policy);

 // Show the Up button in the action bar.

 setupActionBar();

 String[] users={};

 try {

 users = getCloudContent("http://alpreire.appspot.com/getUsers").split("/");

 } catch (IOException e) {

 e.printStackTrace();

 }

 userSpinner = (Spinner) findViewById(R.id.userspinner);

 ArrayAdapter<String> spinnerAdapter = new ArrayAdapter<String>(this,

 R.layout.simple\_spinner\_item, users);

 // spinnerAdapter.setDropDownViewResource(R.id.userspinner);

 userSpinner.setAdapter(spinnerAdapter);

 userSpinner.setSelection(0);

 userSpinner.setOnItemSelectedListener(this);

 setOnClick();

 }

 private String getCloudContent(String add) throws IOException {

 InputStream is = null;

 try {

 URL url = new URL(add);

 HttpURLConnection conn = (HttpURLConnection) url.openConnection();

 conn.setReadTimeout(10000 /\* milliseconds \*/);

 conn.setConnectTimeout(15000 /\* milliseconds \*/);

 conn.setRequestMethod("GET");

 //conn.setInstanceFollowRedirects(false);

 conn.setDoInput(true);

 // Starts the query

 conn.connect();

 is = conn.getInputStream();

 // Convert the InputStream into a string

 String contentAsString = readStream(is);

 return contentAsString;

 // Makes sure that the InputStream is closed after the app is

 // finished using it.

 }catch(IOException e){

 Toast tst = Toast.makeText(this, "Ensure that the device is connected to the internet", Toast.LENGTH\_SHORT);

 tst.show();

 }

 finally {

 if (is != null) {

 is.close();

 }

 }

 return null;

 }

 private String readStream(InputStream in) {

 BufferedReader reader = null;

 StringBuilder sb = new StringBuilder();

 try {

 reader = new BufferedReader(new InputStreamReader(in));

 String line = "";

 while ((line = reader.readLine()) != null) {

 sb.append(line);

 }

 } catch (IOException e) {

 e.printStackTrace();

 } finally {

 if (reader != null) {

 try {

 reader.close();

 } catch (IOException e) {

 e.printStackTrace();

 }

 }

 }

 return sb.toString();

 }

 private void setOnClick() {

 Button saveBtn = (Button) findViewById(R.id.savebtn);

 saveBtn.setOnClickListener(new Button.OnClickListener() {

 public void onClick(View v) {

 Toast tst;

 try {

 if(saveChangesNewUser()){

 tst = Toast.makeText(ctx, "Changes saved", Toast.LENGTH\_SHORT);

 tst.show();

 }

 } catch (IOException e) {

 // TODO Auto-generated catch block

 e.printStackTrace();

 }

 }

 });

 Button newUserBtn = (Button) findViewById(R.id.newUserBtn);

 newUserBtn.setOnClickListener(new Button.OnClickListener() {

 public void onClick(View v) {

 Toast tst;

 try {

 if(saveChangesNewUser()){

 tst = Toast.makeText(ctx, "New User saved", Toast.LENGTH\_SHORT);

 tst.show();

 }

 } catch (IOException e) {

 // TODO Auto-generated catch block

 e.printStackTrace();

 }

 }

 });

 Button deleteBtn = (Button) findViewById(R.id.deletebtn);

 deleteBtn.setOnClickListener(new Button.OnClickListener() {

 public void onClick(View v) {

 String result="";

 try {

 EditText et = (EditText) findViewById(R.id.userid);

 result = getCloudContent("http://alpreire.appspot.com/deleteuser/"+et.getText().toString()).toString();

 Toast tst = Toast.makeText(ctx, result, Toast.LENGTH\_SHORT);

 tst.show();

 } catch (IOException e) {

 // TODO Auto-generated catch block

 e.printStackTrace();

 }

 }

 });

 }

 /\*\*

 \* Set up the {@link android.app.ActionBar}.

 \*/

 private void setupActionBar() {

 getActionBar().setDisplayHomeAsUpEnabled(true);

 }

 @Override

 public boolean onCreateOptionsMenu(Menu menu) {

 // Inflate the menu; this adds items to the action bar if it is present.

 getMenuInflater().inflate(R.menu.user\_mod\_admin, menu);

 return true;

 }

 @Override

 public boolean onOptionsItemSelected(MenuItem item) {

 switch (item.getItemId()) {

 case android.R.id.home:

 // This ID represents the Home or Up button. In the case of this

 // activity, the Up button is shown. Use NavUtils to allow users

 // to navigate up one level in the application structure. For

 // more details, see the Navigation pattern on Android Design:

 //

 // http://developer.android.com/design/patterns/navigation.html#up-vs-back

 //

 NavUtils.navigateUpFromSameTask(this);

 return true;

 }

 return super.onOptionsItemSelected(item);

 }

 @Override

 public void onItemSelected(AdapterView<?> parent, View view, int pos,

 long id) {

 String usrname = (String) parent.getItemAtPosition(pos);

 String[] content={};

 try {

 content = getCloudContent("http://alpreire.appspot.com/getUserContent/" + usrname).split("/");

 EditText et = (EditText) findViewById(R.id.fullname);

 et.setText(content[0]);

 et = (EditText) findViewById(R.id.useraddress);

 et.setText(content[1]);

 et = (EditText) findViewById(R.id.userphone);

 et.setText(content[2]);

 et = (EditText) findViewById(R.id.userid);

 et.setText(content[3]);

 et = (EditText) findViewById(R.id.password);

 et.setText(content[4]);

 } catch (IOException e) {

 // TODO Auto-generated catch block

 e.printStackTrace();

 }

 }

 @Override

 public void onNothingSelected(AdapterView<?> arg0) {

 // TODO Auto-generated method stub

 }

 private boolean saveChangesNewUser() throws IOException{

 boolean success = false;

 String url = "http://alpreire.appspot.com/register/";;

 EditText et = (EditText) findViewById(R.id.fullname);

 url += et.getText().toString() + "/";

 et = (EditText) findViewById(R.id.useraddress);

 url += et.getText().toString() +"/";

 et = (EditText) findViewById(R.id.userphone);

 url += et.getText().toString() +"/";

 et = (EditText) findViewById(R.id.userid);

 url += et.getText().toString().replaceAll("\\s+","") +"/";

 et = (EditText) findViewById(R.id.password);

 url += et.getText().toString() +"/";

 url+= "standard";

 String result="";// = getCloudContent(data);

 try {

 result = getCloudContent(url);

 } catch (Exception e) {

 Toast tst = Toast.makeText(this,"Operation Failed",Toast.LENGTH\_SHORT);

 tst.show();

 }

 if (result.equals("UserCreated") || result.equals("UserUpdated")){

 success = true;

 }

 else{

 success=false;

 }

 return success;

 }

}

# NewVehicle.java

package com.alpr.alpreire;

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStream;

import java.io.InputStreamReader;

import java.net.HttpURLConnection;

import java.net.URL;

import android.app.Activity;

import android.content.Context;

import android.os.Bundle;

import android.os.StrictMode;

import android.support.v4.app.NavUtils;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.view.View.OnClickListener;

import android.widget.AdapterView;

import android.widget.AdapterView.OnItemSelectedListener;

import android.widget.ArrayAdapter;

import android.widget.Button;

import android.widget.EditText;

import android.widget.Spinner;

import android.widget.Toast;

public class NewVehicle extends Activity implements OnItemSelectedListener {

 String destCp;

 Spinner cpSpinner;

 Context ctx = this;

 int makePos=0,colourPos=0,cpPos=0;

 @Override

 protected void onCreate(Bundle savedInstanceState) {

 super.onCreate(savedInstanceState);

 setContentView(R.layout.activity\_new\_vehicle);

 this.destCp = getIntent().getExtras().getString("cpName");

 // Show the Up button in the action bar.

 setupActionBar();

 StrictMode.ThreadPolicy policy = new StrictMode.

 ThreadPolicy.Builder().permitAll().build();

 StrictMode.setThreadPolicy(policy);

 loadSpinners();

 loadCarParks();

 onClicks();

 }

 private void onClicks(){

 Button saveBtn = (Button) findViewById(R.id.saveNV);

 saveBtn.setOnClickListener(new OnClickListener(){

 @Override

 public void onClick(View v) {

 String url = "http://alpreire.appspot.com/newVeh/";

 EditText et = (EditText)findViewById(R.id.nvReg);

 url+= et.getText().toString() + "/";

 Spinner spinner = (Spinner)findViewById(R.id.vehMake);

 url+= spinner.getSelectedItem().toString() +"/";

 et = (EditText)findViewById(R.id.vehMod);

 url+= et.getText().toString() + "/";

 spinner = (Spinner)findViewById(R.id.vehColour);

 url+= spinner.getSelectedItem().toString() +"/";

 et = (EditText)findViewById(R.id.ownerName);

 url+= et.getText().toString() + "/";

 et = (EditText)findViewById(R.id.ownerNum);

 url+= et.getText().toString() + "/";

 et = (EditText)findViewById(R.id.ownerEmail);

 url+= et.getText().toString() + "/";

 spinner = (Spinner)findViewById(R.id.nvCarPark);

 url+= spinner.getSelectedItem().toString();

 try {

 String res = getCloudContent(url);

 Toast tst = Toast.makeText(ctx, res, Toast.LENGTH\_SHORT);

 tst.show();

 } catch (IOException e) {

 // TODO Auto-generated catch block

 e.printStackTrace();

 }

 }

 });

 }

 private void loadCarParks(){

 String[] names={};

 try {

 names = getCloudContent("http://alpreire.appspot.com/getCarParkNames").split("/");

 } catch (IOException e) {

 e.printStackTrace();

 }

 cpSpinner = (Spinner) findViewById(R.id.nvCarPark);

 ArrayAdapter<String> spinnerAdapter = new ArrayAdapter<String>(this,

 R.layout.simple\_spinner\_item, names);

 // spinnerAdapter.setDropDownViewResource(R.id.userspinner);

 cpSpinner.setAdapter(spinnerAdapter);

 cpSpinner.setSelection(0);

 cpSpinner.setOnItemSelectedListener(this);

 }

 private void loadSpinners(){

 Spinner makesSpinner = (Spinner) findViewById(R.id.vehMake);

 // Create an ArrayAdapter using the string array and a default spinner layout

 ArrayAdapter<CharSequence> adapter = ArrayAdapter.createFromResource(this,

 R.array.makes, android.R.layout.simple\_spinner\_item);

 // Specify the layout to use when the list of choices appears

 adapter.setDropDownViewResource(android.R.layout.simple\_spinner\_dropdown\_item);

 // Apply the adapter to the spinner

 makesSpinner.setAdapter(adapter);

 Spinner colourSpinner = (Spinner) findViewById(R.id.vehColour);

 // Create an ArrayAdapter using the string array and a default spinner layout

 ArrayAdapter<CharSequence> colouradapter = ArrayAdapter.createFromResource(this,

 R.array.colours, android.R.layout.simple\_spinner\_item);

 // Specify the layout to use when the list of choices appears

 adapter.setDropDownViewResource(android.R.layout.simple\_spinner\_dropdown\_item);

 // Apply the adapter to the spinner

 colourSpinner.setAdapter(colouradapter);

 }

 /\*\*

 \* Set up the {@link android.app.ActionBar}.

 \*/

 private void setupActionBar() {

 getActionBar().setDisplayHomeAsUpEnabled(true);

 }

 @Override

 public boolean onCreateOptionsMenu(Menu menu) {

 // Inflate the menu; this adds items to the action bar if it is present.

 getMenuInflater().inflate(R.menu.new\_vehicle, menu);

 return true;

 }

 @Override

 public boolean onOptionsItemSelected(MenuItem item) {

 switch (item.getItemId()) {

 case android.R.id.home:

 // This ID represents the Home or Up button. In the case of this

 // activity, the Up button is shown. Use NavUtils to allow users

 // to navigate up one level in the application structure. For

 // more details, see the Navigation pattern on Android Design:

 //

 // http://developer.android.com/design/patterns/navigation.html#up-vs-back

 //

 NavUtils.navigateUpFromSameTask(this);

 return true;

 }

 return super.onOptionsItemSelected(item);

 }

 private String getCloudContent(String add) throws IOException {

 InputStream is = null;

 try {

 URL url = new URL(add);

 HttpURLConnection conn = (HttpURLConnection) url.openConnection();

 conn.setReadTimeout(10000 /\* milliseconds \*/);

 conn.setConnectTimeout(15000 /\* milliseconds \*/);

 conn.setRequestMethod("GET");

 conn.setDoInput(true);

 // Starts the query

 conn.connect();

 is = conn.getInputStream();

 // Convert the InputStream into a string

 String contentAsString = readStream(is);

 return contentAsString;

 // Makes sure that the InputStream is closed after the app is

 // finished using it.

 }catch(IOException e){

 Toast tst = Toast.makeText(this, "Ensure that the device is connected to the internet", Toast.LENGTH\_SHORT);

 tst.show();

 }

 finally {

 if (is != null) {

 is.close();

 }

 }

 return null;

 }

 private String readStream(InputStream in) {

 BufferedReader reader = null;

 StringBuilder sb = new StringBuilder();

 try {

 reader = new BufferedReader(new InputStreamReader(in));

 String line = "";

 while ((line = reader.readLine()) != null) {

 sb.append(line);

 }

 } catch (IOException e) {

 e.printStackTrace();

 } finally {

 if (reader != null) {

 try {

 reader.close();

 } catch (IOException e) {

 e.printStackTrace();

 }

 }

 }

 return sb.toString();

 }

 @Override

 public void onItemSelected(AdapterView<?> parent, View view, int pos, long id) {

 // TODO Auto-generated method stub

 parent.setSelection(pos);

 }

 @Override

 public void onNothingSelected(AdapterView<?> arg0) {

 // TODO Auto-generated method stub

 }

}

# ConvolutionMatrix.java

package com.alpr.alpreire.imagemanipulation;

import android.graphics.Bitmap;

import android.graphics.Color;

public class ConvolutionMatrix {

 public static final int SIZE = 3;

 public double[][] Matrix;

 public double Factor = 1;

 public double Offset = 1;

 //Constructor with argument of size

 public ConvolutionMatrix(int size) {

 Matrix = new double[size][size];

 }

 public void setAll(double value) {

 for (int x = 0; x < SIZE; ++x) {

 for (int y = 0; y < SIZE; ++y) {

 Matrix[x][y] = value;

 }

 }

 }

 public void applyConfig(double[][] config) {

 for(int x = 0; x < SIZE; ++x) {

 for(int y = 0; y < SIZE; ++y) {

 Matrix[x][y] = config[x][y];

 }

 }

 }

 public static Bitmap computeConvolution3x3(Bitmap src, ConvolutionMatrix matrix) {

 int width = src.getWidth();

 int height = src.getHeight();

 Bitmap result = Bitmap.createBitmap(width, height, src.getConfig());

 int A, R, G, B;

 int sumR, sumG, sumB;

 int[][] pixels = new int[SIZE][SIZE];

 for(int y = 0; y < height - 2; ++y) {

 for(int x = 0; x < width - 2; ++x) {

 // get pixel matrix

 for(int i = 0; i < SIZE; ++i) {

 for(int j = 0; j < SIZE; ++j) {

 pixels[i][j] = src.getPixel(x + i, y + j);

 }

 }

 // get alpha of center pixel

 A = Color.alpha(pixels[1][1]);

 // init color sum

 sumR = sumG = sumB = 0;

 // get sum of RGB on matrix

 for(int i = 0; i < SIZE; ++i) {

 for(int j = 0; j < SIZE; ++j) {

 sumR += (Color.red(pixels[i][j]) \* matrix.Matrix[i][j]);

 sumG += (Color.green(pixels[i][j]) \* matrix.Matrix[i][j]);

 sumB += (Color.blue(pixels[i][j]) \* matrix.Matrix[i][j]);

 }

 }

 // get final Red

 R = (int)(sumR / matrix.Factor + matrix.Offset);

 if(R < 0) { R = 0; }

 else if(R > 255) { R = 255; }

 // get final Green

 G = (int)(sumG / matrix.Factor + matrix.Offset);

 if(G < 0) { G = 0; }

 else if(G > 255) { G = 255; }

 // get final Blue

 B = (int)(sumB / matrix.Factor + matrix.Offset);

 if(B < 0) { B = 0; }

 else if(B > 255) { B = 255; }

 // apply new pixel

 result.setPixel(x + 1, y + 1, Color.argb(A, R, G, B));

 }

 }

 // final image

 return result;

 }

 }

# buzzers.py

from \_\_future\_\_ import print\_function # GAE.

# GAE is a READ-ONLY filesystem!!!! (Godammit, Jim...).

from functools import wraps

from flask import Flask, render\_template, request, redirect, url\_for, session

from google.appengine.ext import db

#\_VALIDUSERS = 'data/validusers.txt'

#\_WAITING = 'data/waiting.txt'

# webapp = Flask('\_\_name\_\_')

webapp = Flask(\_\_name\_\_.split('.')[0]) # GAE.

# class for user Account

class UserAccount(db.Model):

 username = db.StringProperty()

 password = db.StringProperty()

 usertype = db.StringProperty()

 useraddress= db.StringProperty()

 fullname = db.StringProperty()

 userphone = db.StringProperty()

class CarPark(db.Model):

 cpName = db.StringProperty()

 cpAddress = db.StringProperty()

 cpCapacity = db.StringProperty()

 cpAttendant = db.StringProperty()

class Vehicle(db.Model):

 registration = db.StringProperty()

 Make = db.StringProperty()

 Model = db.StringProperty()

 Colour = db.StringProperty()

 ownerName = db.StringProperty()

 ownerNum = db.StringProperty()

 ownerEmail = db.EmailProperty()

 carPark = db.StringProperty()

admin = UserAccount(username = 'admin',password='admin', fullname= "John Doe", usertype = 'admin', useraddress="123 Park Ave", userphone="0987654321")

admin.put()

carPark= CarPark(cpName="ITCarlow", cpAddress="Kilkenny Road, Carlow", cpCapacity="500", cpAttendant="admin")

carPark.put()

vehicle = Vehicle(registration="132ls277", Make="Skoda", Model="Octavia", Colour="Black", ownerName="Joe Bloggs", ownerNum="8512312312", ownerEmail="asdf@mail.com")

vehicle.put()

def checkOKtype(utype):

 """ This (parameterised) decorator makes sure the user can only look at the

 content that's meant for them (unless they are of type 'admin', in which

 case they can see EVERYTHING).

 """

 def check\_usertype(func):

 @wraps(func)

 def wrapped\_function(\*args, \*\*kwargs):

 if session['usertype'] in (utype, 'admin'):

 return(func(\*args, \*\*kwargs))

 else:

 return redirect(url\_for('indexpage'))

 return wrapped\_function

 return check\_usertype

@webapp.route('/')

def indexpage():

 return "IndexPage"

#user database interface

@webapp.route('/login/<username>/<passwd>', methods=["GET", "POST"])

def dologin(username, passwd):

 """ Either display the login form, or process a filled-in login

 form. Only authorized users can login.

 """

 def validusertype(u2check, p2check):

 if db.GqlQuery("SELECT \* FROM UserAccount WHERE username = :u2check AND password = :passwd", u2check = u2check, passwd=p2check).count() > 0:

 qry = db.GqlQuery("SELECT \* FROM UserAccount WHERE username = :u2check", u2check = u2check)

 for r in qry.fetch(limit=None):

 return r.usertype

 return None

 if request.method == "GET":

 \_type = validusertype(username, passwd)

 if \_type:

 return "True"

 # elif request.method == "POST":

 # Fall though - if nothing above takes, we end up here.

 return "False"

@webapp.route('/register/<fullname>/<address>/<phonenum>/<username>/<passwd>/<usertype>')

def doregistration(fullname, address, phonenum, username, passwd, usertype):

 def check\_in\_db(u2check):

 if db.GqlQuery("SELECT \* FROM UserAccount WHERE username = :u2check", u2check = u2check).count() > 0:

 return 1

 else:

 return 0

 if check\_in\_db(username) == 0:

 newUser = UserAccount(username = username,password=passwd, fullname= fullname, usertype = usertype, useraddress=address, userphone=phonenum)

 newUser.put()

 res="UserCreated"

 return res

 elif check\_in\_db(username) == 1:

 return "UserUpdated"

 else:

 return "ErrorInReg"

@webapp.route("/getUsers")

def getUsers():

 qry = db.GqlQuery("SELECT \* FROM UserAccount")

 names=""

 for r in qry.fetch(limit=None):

 names += r.username + "/"

 return names

@webapp.route("/getUserContent/<username>")

def getUserContent(username):

 qry = db.GqlQuery("SELECT \* FROM UserAccount WHERE username = :username", username=username)

 content = ""

 for r in qry.fetch(limit=None):

 content="""{}/{}/{}/{}/{}/{}""".format(r.fullname, r.useraddress, r.userphone, r.username ,r.password, r.usertype)

 return content

@webapp.route('/deleteuser/<userid>')

def deleteuser(userid):

 def check\_in\_db(u2check):

 if db.GqlQuery("SELECT \* FROM UserAccount WHERE username = :u2check", u2check = u2check).count() > 0:

 return 1

 else:

 return 0

 if request.method == "GET":

 if check\_in\_db(userid) == 1:

 q = db.GqlQuery("SELECT \* from UserAccount where username = :userid", userid=userid)

 result=q.fetch(10)

 for res in result:

 db.delete(resd)

 return "UserDeleted"

 elif check\_in\_db(userid) == 0:

 return "UserNotFound"

 else:

 return "ErrorInDel"

 return "Error"

#car park databse interface

@webapp.route("/newcarpark/<name>/<address>/<capacity>/<attendant>")

def newUpdateCarPark(name, address, capacity,attendant):

 def check\_in\_db(cpCheck):

 if db.GqlQuery("SELECT \* FROM CarPark WHERE name = :cpCheck", cpCheck = cpCheck).count() > 0:

 return 1

 else:

 return 0

 if check\_in\_db(name) == 0:

 newCp = CarPark(cpName=name, cpAddress=address,cpCapacity=capacity,cpAttendant=attendant)

 newCp.put()

 res = "CarParkCreated"

 return res

 elif check\_in\_db(name) == 1:

 res="CarParkUpdated"

 return res

 else:

 res="ErrorInCarParkEdit"

 return res

@webapp.route("/getCarParkNames")

def getCarParkNames():

 qry = db.GqlQuery("SELECT \* FROM CarPark")

 names=""

 for r in qry.fetch(limit=None):

 names += r.cpName + "/"

 return names

@webapp.route("/getCPContent/<name>")

def getCPContent(name):

 qry = db.GqlQuery("SELECT \* FROM CarPark WHERE cpName = :cname", cname=name)

 content = ""

 for r in qry.fetch(limit=None):

 content="""{}/{}/{}/{}""".format(r.cpName, r.cpAddress, r.cpCapacity, r.cpAttendant)

 return content

@webapp.route('/newVeh/<regNo>/<make>/<model>/<colour>/<ownerName>/<ownerNum>/<ownerEmail>/<carPark>')

def newVehicle(regNo, make, model,colour,ownerName, ownerNum, ownerEmail, carPark):

 def check\_in\_db(reg, cp):

 if db.GqlQuery("SELECT \* FROM Vehicle WHERE registration = :reg and carPark=:cp", reg = reg, cp=cp).count() > 0:

 return 1

 else:

 return 0

 if check\_in\_db(regNo, carPark) == 0:

 newCar = Vehicle(registration=regNo, Make=make, Model=model, Colour=colour, ownerName=ownerName, ownerNum=ownerNum, ownerEmail=ownerEmail, carPark=carPark)

 newCar.put()

 res = "Vehicle added to {}".format(carPark)

 return res

 elif check\_in\_db(regNo, carPark)==1:

 return "{} already Exists".format(regNo)

 else:

 return "Failed To add vehicle"

@webapp.route("/checkcar/<reg>/<carPark>")

def carcheck(reg, carPark):

 if db.GqlQuery("SELECT \* FROM Vehicle WHERE registration = :reg and carPark=:cp", reg = reg, cp=carPark).count() > 0:

 return "Vehicle is Registered"

 else:

 return "Vehicle Unregistered"

webapp.secret\_key = b'youwillneverguessmysecretkeyhahahahahahaaaaaaa' # GAE.

if \_\_name\_\_ == '\_\_main\_\_':

 webapp.run(debug=True, host='0.0.0.0')